(19)日本国特許庁（ＪＰ）　(12)公表特許公報（Ａ）　(11)特許出願公表番号
特表2002－517132
（P2002－517132A）
(43)公表日　平成14年６月11日(2002.6.11)

(51)Int.Cl.⁷	識別記号	ＦＩ		テーマコード（参考）
Ｈ０４Ｌ　7/00		Ｈ０４Ｌ　7/00	Ｂ	５Ｋ０２８
			Ｚ	５Ｋ０４７
Ｈ０４Ｊ　3/00		Ｈ０４Ｊ　3/00	Ｈ	
Ｈ０４Ｌ　12/28		Ｈ０４Ｌ　11/00	３１０Ｄ	

審査請求　未請求　予備審査請求　有　（全 30 頁）

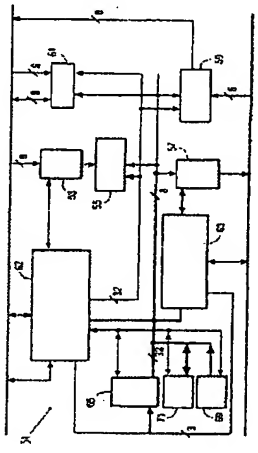
(21)出願番号　特願2000－551514(P2000－551514)
(86)(22)出願日　平成11年５月20日(1999.5.20)
(85)翻訳文提出日　平成12年１月26日(2000.1.26)
(86)国際出願番号　ＰＣＴ／ＩＢ９９／００９２８
(87)国際公開番号　ＷＯ９９／６２２１６
(87)国際公開日　平成11年12月２日(1999.12.2)
(31)優先権主張番号　０９／０８６，２７０
(32)優先日　平成10年５月28日(1998.5.28)
(33)優先権主張国　米国（ＵＳ）
(81)指定国　ＥＰ(ＡＴ，ＢＥ，ＣＨ，ＣＹ，ＤＥ，ＤＫ，ＥＳ，ＦＩ，ＦＲ，ＧＢ，ＧＲ，ＩＥ，ＩＴ，ＬＵ，ＭＣ，ＮＬ，ＰＴ，ＳＥ)，ＪＰ

(71)出願人　コーニンクレッカ　フィリップス　エレクトロニクス　エヌ　ヴィ
Ｋｏｎｉｎｋｌｉｊｋｅ　Ｐｈｉｌｉｐｓ　Ｅｌｅｃｔｒｏｎｉｃｓ　Ｎ．Ｖ．
オランダ国　5621　ベーアー　アインドーフェン　フルーネヴァウツウェッハ　1
(72)発明者　ハルヤルカー，サミール　エヌ
オランダ国，5656　アーアー　アインドーフェン，プロフ・ホルストラーン　6
(72)発明者　シェクター，エリク
オランダ国，5656　アーアー　アインドーフェン，プロフ・ホルストラーン　6
(74)代理人　弁理士　伊東　忠彦

最終頁に続く

(54)【発明の名称】　予約ベースＴＤＭＡプロトコルのタイムスタンプ同期方法

(57)【要約】
媒体アクセス制御サブシステム（例えば予約ベースＴＤＭＡプロトコルを用いるもの）によって媒介される共通チャネルを通じて相互に通信する制御ノードと複数の他のノードとを含むネットワーク（例えば無線ＡＴＭネットワーク）内でタイムスタンプを同期させる方法である。方法は、第１の時間において制御ノードから第１の命令を送信する段階と、第１の命令に応じて、制御ノード及び各他のノードの中のレジスタの中に開始タイムスタンプを記憶する段階と、第１の時間よりも遅い第２の時間において制御ノードから第２の命令を送信する段階とを含む。各他のノードは開始タイムスタンプ値と現在タイムスタンプ値との間の差を計算し、計算された差を現在タイムスタンプ値に加算する。他の実施例では、制御ノードはプリセット命令を送信し、各他のノードは上記プリセット命令に応じて夫々のタイムスタンプの値を指示された初期タイムスタンプ値にプリセットする。各他のノードは現在のタイムスタンプカウンタ値を指示されたリセット値と周期的に比較し、一致が検出されると、次のプリセット値に応じて次のタイムスタンプ値補



【特許請求の範囲】

【請求項１】　媒体アクセス制御サブシステムによって媒介される共通チャネルを通じて相互に通信する制御ノードと複数の他のノードとを含むネットワーク内でタイムスタンプを同期させる方法であって、

第１の時間において制御ノードから第１の命令を送信する段階と、

第１の命令に応じて、制御ノード及び各他のノードの中のレジスタの中に開始タイムスタンプを記憶する段階と、

第１の時間よりも遅い第２の時間において制御ノードから第２の命令を送信する段階と、

各他のノードが開始タイムスタンプ値と現在タイムスタンプ値との間の差を計算し、計算された差を現在タイムスタンプ値に加算する段階とを含む方法。

【請求項２】　媒体アクセス制御システムは予約ベースＴＤＭＡプロトコルを使用する、請求項１記載の方法。

【請求項３】　第１の命令はタイムスタンプ獲得命令を含み、第２の命令はタイムスタンプロード命令を含む、請求項１記載の方法。

【請求項４】　ネットワーク内のノードのうち任意の選択された１つのノードは、異なる時間において制御ノードとして動作するよう割り当てられ得る、請求項１記載の方法。

【請求項５】　媒体アクセス制御サブシステムによって媒介される共通チャネルを通じて相互に通信する制御ノードと複数の他のノードとを含むネットワーク内でタイムスタンプを同期させる方法であって、

制御ノードからプリセット命令を送信する段階と、

各他のノードが上記プリセット命令に応じて夫々のタイムスタンプの値を指示された初期タイムスタンプ値にプリセットする段階と、

各他のノードが現在のタイムスタンプカウンタ値を指示されたリセット値と周期的に比較し、夫々の現在のタイムスタンプカウンタ値と指示されるリセット値との間の一致が検出されると、次のプリセット値に応じて次のタイムスタンプ値補正が行われるべきであることを示す同期フラグラッチを設定する段階とを含む方法。

【請求項６】　媒体アクセス制御サブシステムは予約ベースＴＤＭＡプロトコルを使用する、請求項５記載の方法。

【請求項７】　ネットワーク内のノードのうち任意の選択された１つのノードは、異なる時間において制御ノードとして動作するよう割り当てられ得る、請求項５記載の方法。

【請求項８】　制御ノードと、

複数の他のノードと、

制御ノード及び複数の他のノードによって共通に使用される通信チャネルへのアクセスを媒介する媒体アクセス制御サブシステムとを含むネットワークであって、

制御ノード及び複数の他のノードは夫々、

指定された伝送時間において制御ノードによって伝送される制御パケットのプリセット命令バイトを記憶する比較器レジスタと、

プリセット命令バイトの値と指示されたヘッダ値とを比較し、一致が検出されるとトリガ信号を出力する第１の比較器と、

タイムスタンプカウンタと、

タイムスタンプレジスタと、

同期フラグラッチと、

トリガ信号に応じて、タイムスタンプカウンタの中に記憶されたタイムスタンプ値を読出し、読み出されたタイムスタンプ値をタイムスタンプレジスタの中へロードし、タイムスタンプカウンタを指示されたプリセット値でプリセットする状態機械と、

タイムスタンプによって記憶された現在のタイムスタンプ値を指示されたリセット値と比較し、一致が検出されると、次のプリセット命令バイトに応じて次のタイムスタンプ値補正が行われることを示すため同期フラグラッチを設定する第２の比較器とを含むネットワーク。

【請求項９】　制御ノードのための指示されたプリセット値はゼロである、請求項８記載のネットワーク。

【請求項１０】　各他のノードのための指示されたプリセット値は物理層を

通じた遅延に対応する、請求項８又は９記載のネットワーク。

【請求項１１】　媒体アクセス制御サブシステムは予約ベースＴＤＭＡプロトコルを使用する、請求項８記載のネットワーク。

【請求項１２】　指定された伝送時間は制御データフレームのＢＳ＿ＳＩＧフェーズに対応する、請求項８記載のネットワーク。

【請求項１３】　ネットワーク内のノードのうち任意の選択された１つのノードは、異なる時間において制御ノードとして動作するよう割り当てられ得る、請求項８記載のネットワーク。

【請求項１４】　制御ノードと、

複数の他のノードと、

制御ノード及び複数の他のノードによって共通に使用される通信チャネルへのアクセスを媒介する媒体アクセス制御サブシステムとを含むネットワークであって、

制御ノード及び複数の他のノードは夫々、

制御ノードによって伝送される制御パケットの命令バイトを記憶する比較器レジスタと、

プリセット命令バイトの値と指示されたヘッダ値とを比較し、一致が検出されるとトリガ信号を出力する第１の比較器と、

タイムスタンプカウンタと、

タイムスタンプレジスタと、

同期フラグラッチと、

オフセットレジスタと、

トリガ信号に応じて、タイムスタンプカウンタの中に記憶されたタイムスタンプ値を読出し、読み出されたタイムスタンプ値をタイムスタンプレジスタの中へロードする状態機械と、

第１の時間よりも遅い第２の時間において制御ノードによって伝送された最後のタイムスタンプ値とタイムスタンプレジスタの中に記憶される現在のタイムスタンプ値との間の差を計算し、この差をオフセット値としてオフセットレジスタの中へ記憶するプロセッサ回路と、

オフセットレジスタの中に記憶されたオフセット値をタイムスタンプカウンタの現在値へ加算し、その和をタイムスタンプカウンタの中へ再ロードする、加算器と、

タイムスタンプによって記憶された現在のタイムスタンプ値を指示されたリセット値と比較し、一致が検出されると、次のプリセット命令バイトに応じて次のタイムスタンプ値補正が行われることを示すため同期フラグラッチを設定する第２の比較器とを含むネットワーク。

【請求項１５】　制御ノードによって伝送された最後のタイムスタンプ値は、前の命令バイトが伝送された時間において制御ノードのタイムスタンプレジスタの中に記憶されるタイムスタンプ値に対応する、請求項１４記載のネットワーク。

【請求項１６】　媒体アクセス制御サブシステムは、予約ベースＴＤＭＡプロトコルを使用する、請求項１４記載のネットワーク。

【請求項１７】　ネットワーク内のノードのうち任意の選択された１つのノードは、異なる時間において制御ノードとして動作するよう割り当てられ得る、請求項１４記載のネットワーク。

【発明の詳細な説明】

【０００１】

本発明は概してネットワーク内で同一チャネルの多重アクセスを可能とするシステム及び方法に関連し、更に特定的には、予約ベースＴＤＭＡプロトコルを使用するネットワーク内のタイムスタンプ同期のシステム及び方法に関連する。

【０００２】

概して、特に無線ネットワークといった通信ネットワークは典型的には、同一チャネルを使用するネットワーク内の多数の送信器によるデータパケットの同時伝送によるデータパケットの衝突を防止するよう設計される多重アクセスプロトコルを使用する。広く用いられるようになったプロトコルの１つには、時分割多重アクセス（ＴＤＭＡ）として知られるものがある。この技術の詳細な説明は、参考文献「Telecommunications Networks: Protocols, Modeling and Analysis, Addision-Wesley, 1997」に記載されている。概して、ＴＤＭＡプロトコルによれば、チャネル時間は小さな時間スロットへ分割され、各時間スロットは異なるノード（ユーザ）へ割り当てられる。この時間スロット割当ては、固定であってもよく（従来のＴＤＭＡ）、又は可変であってもよい（予約ベースＴＤＭＡ）。いずれの場合も、ノード（ユーザ）数は限られているため、データは通常はＴＤＭＡ「フレーム」の中で伝送され、これは異なるユーザが受ける遅延が有限であることを確実とする。

【０００３】

例えば、固定割当てＴＤＭＡでは、ＴＤＭＡフレームは全てのユーザに割り当てられたスロットの総数を含み、その後にＴＤＭＡフレームが繰り返される。予約ベースＴＤＭＡの場合、自然なフレーム割当てはＴＤＭＡフレームの異なる「フェーズ」に関して行われ、これらのフェーズは典型的には、予約が要求され割り当てられる「制御」フェーズ、異なるユーザによって夫々に割り当てられたタイムスロットの中でデータが伝送される「データ」フェーズとを含む。

【０００４】

ＴＤＭＡネットワーク内の全ての送信器及び受信器はＴＤＭＡフレームに関して同期されることが必要である。誤って同期された送受信器は、最も良い場合で

も通信できず、最悪の場合はプロトコルの中に適当なセーフガードが構築されていなければＴＤＭＡネットワーク全体を崩壊させうる。ＴＤＭＡフレーム同期は、物理層の機能であるモデムのクロック同期と同じでないことが認識されねばならない。通常、フレーム同期は中央制御器（ＣＣ）によって実施される中央集中制御方式によって達成されうる。しかしながら、フレーム同期は分散された方式でも実施されうる。

【０００５】

殆どのＴＤＭＡネットワークでは、伝送用の資源を正しく割り当てるために普遍的な時間基準が必要とされる。この普遍的な時間基準は通常は例えば午後２時といった現在時間を特定する「タイムスタンプ」の形式で与えられる。タイムスタンプは、中央制御器によって周期的にブロードキャストされ、最終端末（ＥＴ）においてそれらの「タイムスタンプ」レジスタを同期させるために使用される。

【０００６】

可変長のＴＤＭＡフレームでは、タイムスタンプを使用することによって達成される同期は、典型的には各ＥＴ中の位相ロックループ（ＰＬＬ）の使用を必要とし、これはかなり複雑である。更に、このために使用されるＰＬＬはタイムスタンプスキームのパラメータが変更されるたびに、例えばタイムスタンプ伝送の周波数が変化されるたびに再設計されねばならない。これに関して、ＥＴが多くの異なるネットワークにおいて互換可能に使用されることを可能とするため、包括的な同期スキームが必要とされる。

【０００７】

現在公知のフレーム同期技術では、タイムスタンプ値は共通のクロックから発生される。例えば、ＩＥＥＥ１３９４標準によれば、ローカル１３９４シリアルバスの中でのクロック分布は、現在のｂｕｓ＿ｔｉｍｅ（即ちタイムスタンプ）を含むｃｙｃｌｅ＿ｓｔａｒｔ（サイクル開始）パケットを周期的に送信するサイクルマスター（即ち中央制御器）によって達成される。この「再同期」は理想的には１２５ｍｓ毎に行われる。しかしながら、ｃｙｃｌｅ＿ｓｔａｒｔパケットの伝送は、ｃｙｃｌｅ＿ｓｔａｒｔパケットが送信される必要があるときにロ

ーカル１３９４シリアルバスが他のノードによって使用されており、それによりｃｙｃｌｅ＿ｓｔａｒｔパケットを送信する前にノードがその伝送を終了するまでサイクルマスターが待つことを要求するため、僅かに遅延されうる。

【０００８】

ここで図１を参照するに、１３９４サイクルマスターにおいてｃｙｃｌｅ＿ｓｔａｒｔパケットを発生する方法を説明する。更に特定的には、２４．５７６ＭＨｚのマスタクロックレートで動作する水晶発振器２０の出力は、サイクルカウンタ２２へ入力される。フレーム同期は、ローカル１３９４シリアルバスによって相互接続される全ての１３９４ノードの中のサイクルカウンタを同期させることによって達成される。サイクルカウンタ２２の出力はそれに応じて１２５ｍｓ毎に状態機械２６へトリガ信号を送信するモジュロ１２５ｍｓブロック２４へ渡される。

【０００９】

状態機械２６は、モジュロ１２５ｍｓブロック２４によって送信されるトリガ信号に応じて１３９４物理（ＰＨＹ）層２８へチャネル要求信号を送信する。チャネルが使用可能になったとたんに、１３９４ＰＨＹ層２８は状態機械２６へチャネル使用可能信号を送信する。チャネル使用可能信号に応じて、状態機械２６はｃｙｃｌｅ＿ｓｔａｒｔパケット用のパケットヘッダを準備し、現在のｂｕｓ＿ｔｉｍｅ（タイムスタンプ値）を発生するよう正しい時点においてサイクルカウンタ２２の内容をラッチするためレジスタ３０へイネーブル信号を送信する。処理における全ての遅延は、レジスタ３０へのイネーブル信号を処理遅延の量によって正しく遅延させることによって容易に考慮に入れられ得る。

【００１０】

受信器（即ちｃｙｃｌｅ＿ｓｔａｒｔパケットを受信する各ノード）において、そのノードの中のサイクルカウンタはｃｙｃｌｅ＿ｓｔａｒｔパケットを介して受信された適当なｂｕｓ＿ｔｉｍｅに設定されねばならない。このタスクを達成するための方法を、図２を参照して説明する。更に特定的には、受信器ノードのＰＨＹ層３３は、ｃｙｃｌｅ＿ｓｔａｒｔパケットを受信し、これをｃｙｃｌｅ＿ｓｔａｒｔパケットが実際にｃｙｃｌｅ＿ｓｔａｒｔパケットであることを

確認するために復号化するｃｙｃｌｅ＿ｓｔａｒｔヘッダ復号化ブロック３５を含むリンク層へ送信する。同時に、ｂｕｓ＿ｔｉｍｅ値はｃｙｃｌｅ＿ｓｔａｒｔパケットから抽出され、レジスタ３７へロードされる。処理遅延ブロック３９は、（復号化動作及び／又はｂｕｓ＿ｔｉｍｅ値をレジスタ３７へロードするための）任意の処理遅延を、レジスタ３７の出力へ、又はｃｙｃｌｅ＿ｓｔａｒｔヘッダ復号化ブロック３５によって出力されるロード信号へ加える。ロード信号は、レジスタ３７の内容の和をロードすること及びサイクルカウンタ４１の中へ遅延を処理することを可能とする受信器ノードのサイクルカウンタ４１のロード端子へ印加される。

【００１１】

サイクルカウンタ４１をリセットする間にサイクルが正確であることが非常に重要であることが認識されるべきであり、即ち、サイクルマスターによって伝送されるｂｕｓ＿ｔｉｍｅがｃｙｃｌｅ＿ｓｔａｒｔパケットが送信される正確な時間に対応せねばならず、各受信器ノードにおける処理遅延が非常に精密に決定されねばならないことを意味する。

【００１２】

１２５ｍｓ毎にサイクルカウンタ４１をリセットすることは、異なる水晶発振器から得られるクロックが相互に大きくドリフトしないことを確実とする。殆どのプロトコルは、その間にタイムスタンプ更新が送信されねばならない間隔を有する。そうでなければ、タイミングジッタは特定の適用、例えばＭＰＥＧ復号化器によって扱われうるものよりも大きくなる。

【００１３】

ＩＥＥＥ８０２．１１は同様のプロトコルを有する。更に特定的には、ＩＥＥＥ８０２．１１標準によれば、ＩＥＥＥ１９３４標準によるｃｙｃｌｅ＿ｓｔａｒｔパケットのブロードキャストと同様に、「ビーコン」が周期的にブロードキャストされる。ビーコンは、他のフィールドと共に、タイムスタンプ値及びタイムスタンプ間隔を含む。タイムスタンプ間隔は、ＩＥＥＥ１３９４標準では１２５ｍｓサイクル時間に固定されているのに対して、ＩＥＥＥ８０２．１１標準では可変である。ＩＥＥＥ１３９４標準と同様、チャネルがすぐに使用可能でなけ

れば、ビーコンの伝送は遅延されうる。ＩＥＥＥ１３９４と同様、タイムスタンプ値がビーコンパケットの伝送の正確な時間に対応せねばならないことが主な要件である。

【００１４】

ＩＥＥＥ１３９４及びＩＥＥＥ８０２．１１は共に予約ベースプロトコルでないことが認識されうるべきである。両方の場合、まず特定の受信器ノードのためのアクセスを決定するためにアービトレーションフェーズが開始される。ｃｙｃｌｅ＿ｓｔａｒｔ及びビーコンパケットでさえ、まず伝送の前にチャネルの使用のアービトレーションを行わねばならない。

【００１５】

予約ベースＴＤＭＡプロトコルでは、タイムスタンプベースアプローチについて多くの問題がある。第１の問題は、タイムスタンプ値の伝送もまた予約されねばならず、続いて、他のデータもまた伝送のためにキューに入れられねばならないことである。（多くの他の機能を管理するために使用されねばならない）プロセッサ資源の有効な使用を確実とするため、このキュー動作は通常は予めスケジューリングされる。しかしながら、タイムスタンプ値は伝送の正確な時間まで得られない。更に、タイムスタンプ値の後のデータパケット値をキューに入れることは、タイムスタンプ値が獲得される前には行われ得ない。もちろん、データストリームを２つの別個のキュー、即ち一方はタイムスタンプ値を保持するキュー、他方はデータを保持するキュー、の間で切り換えることが可能である。しかしながら、この解法はかなり複雑であり、正確な同期を必要とする。

【００１６】

この問題のより詳細な理解は、予約ベース媒体アクセス制御（ＭＡＣ）プロトコルを使用する無線非同期転送モード（ＡＴＭ）ネットワークの場合を考慮することによって得られる。ＭＡＣプロトコル実施は、C.Y.Ngo及びS.N.Hulyalkarによる「A Detailed MAC Sub-System Description for BS-Oriented WATM LAN@, WP3, D3.1, Aug. 15, 1996」に記載されるように、周期的制御データフレーム（ＣＤＦ）に依存する。各ＣＤＦは、多くのフェーズを含み、この間に制御及びデータ情報の両方は基地局（ＢＳ）から無線端末（ＷＴ）へ送信される。図３は、

かかる一般的な構造を実施するための４つのフェーズ、即ちＢＳ＿ＳＩＧ；ＤＮ＿ＤＡＴＡ；ＵＰ＿ＤＡＴＡ；及びＥ＿ＢＵＲＳＴを含む。これらのフェーズの夫々の簡単な説明は以下の通りである。

【００１７】

ＢＳ＿ＳＩＧ：このフェーズ中、ＢＳはダウンリンク用の制御情報を送信する。タイムスタンプパケットはこのフェーズ中に送信されると想定される。ＢＳにおいて、プロセッサはＢＳからパケットの伝送を開始する。ＷＴにおいて、ＷＴはＢＳからのパケットの受信のプロセスを開始する。

【００１８】

ＤＮ＿ＤＡＴＡ：このフェーズ中、ＢＳはＷＴ用のデータパケットを送信する。ＢＳにおいて、プロセッサはＵＰ＿ＤＡＴＡフェーズ中にＷＴによって送信されたパケットを解釈している。ＷＴにおいて、プロセッサはＵＰ＿ＤＡＴＡフェーズ中の伝送の次のバーストのためのＰＨＹ　ＦＩＦＯを記憶している。

【００１９】

ＵＰ＿ＤＡＴＡ：このフェーズ中、ＷＴはＢＳ用のデータ及びシグナリングパケットを送信する。シグナリングは、スーパースロットを用いて送信される。ＢＳにおいて、プロセッサはＢＳ＿ＳＩＧ及びＤＮ＿ＤＡＴＡフェーズ中の伝送の次のバーストのためのＰＨＹ　ＦＩＦＯを記憶している。ＷＴにおいて、プロセッサはＢＳ＿ＳＩＧ及びＤＮ＿ＤＡＴＡフェーズ中にＢＳによって送信されるパケットを解釈している。

【００２０】

Ｅ＿ＢＵＲＳＴ：このフェーズ中、ＵＰ＿ＤＡＴＡフェーズ中の伝送用に空間を現在割り当てられていないＷＴは、ＷＡＴＭネットワークに入りたいか否かを示す。ＷＴ及びＢＳのプロセッサは共に、Ｅ＿ＢＵＲＳＴフェーズを実施している。

【００２１】

ハードウエア設計は、ＣＤＦの４つのフェーズを計算するための基礎としての同一のタイムスタンプ値を維持するＢＳ及び各ＷＴに基づく。パケットを効率的に通信し転送するために、全ては同じ時間期間を維持せねばならない。全ては、

基地局の値を複製することによってそれらのタイムスタンプ値を周期的に同期させねばならず、全てはＢＳから開始時間命令を得なければならない。

【００２２】

ＭＡＣプロセッサは、ＷＴ及びＢＳの両方のために割込駆動されることが想定される。ＢＳはシステム全体のタイミングを決定する。タイムスタンプ値を基準として使用して、各フェーズが動作するときの正確な時間を決定する。このタイミング情報は、ＢＳ＿ＳＩＧＮ＿ｎフェーズ中に送信される。全てのフェーズは夫々に連続するため、ＷＴ及びＢＳはタイミング情報に基づいて次のフェーズのためのカウンタを設定し、これは次にカウンタがあふれたときにプロセッサに対する割込みをトリガする。プロセッサは割り当てられた時間中の夫々のフェーズのフェーズの間にその機能を終了し、次のフェーズのために準備ができていなくてはならない。

【００２３】

タイムスタンプ同期のために、ＢＳはＢＳ＿ＳＩＧフェーズ中にタイムスタンプ値を送信することが想定されうる。しかしながら、ＢＳは、ＰＨＹ＿ＦＩＦＯに、ＢＳ＿ＳＩＧ及びＤＢ＿ＤＡＴＡフェーズ中の伝送用のパケットを記憶していることに注意すべきである。しかしながら、タイムスタンプ値はＢＳ＿ＳＩＧフェーズ中に決定されねばならず、ＵＰ＿ＤＡＴＡフェーズ中は獲得されえない。従って、通常の伝送ストリームは、タイムスタンプ値が伝送時間中にタイムスタンプレジスタからロードされることを可能とするために停止されねばならない。この解法は、直接データパスと競合するため望ましくない。

【００２４】

上述の問題は考慮される特定のプロトコルによるものでなく、概してプロトコルの予約ベース特性によるものであり、それにより特定の時間において何が伝送されるべきかに関する決定はこれらの時間より前もって行われねばならないことが認識されるべきである。

【００２５】

この問題に対する１つの解法は、ＢＳ＿ＳＩＧフェーズ中にタイムスタンプパケットのみを送信することである。しかし、５３バイトセルを用いる無線ＡＴＭ

では、タイムスタンプ値はたった４バイトでありうるため、これは資源の無駄である。更に一般的には、タイムスタンプ情報のみの伝送のために固定スロットを使用することはネットワーク資源の無駄をもたらす。これに関して、「最小の」スロット割当てなしではチャネル時間を予約することが可能でないため、この固定スロットは全ての予約ベースプロトコルの本質的な面であることが認識されるべきである。最小の時間スロットがタイムスタンプ値のみを送信するのに要求される時間よりもまだかなり大きいことにより、問題が生ずる。

【００２６】

上述のことより、予約ベースＴＤＭＡプロトコルネットワークにおけるタイムスタンプ同期の方法のために、上述の欠点及び現在利用可能な技術の短所を克服する技術が必要とされることが認められ得る。本発明はこの技術におけるこの必要性を満たす。

【００２７】

本発明は、媒体アクセス制御サブシステム（例えば予約ベースＴＤＭＡプロトコルを用いたもの）によって媒介される共通のチャネルを通じて相互に通信する制御ノード及び複数の他のノードを含むネットワーク（例えば無線ＡＴＭネットワーク）内でタイムスタンプを同期させる方法に関する。方法は、第１の時間において制御ノードから第１の命令を送信する段階と、第１の命令に応じて、制御ノード及び各他のノードの中のレジスタの中に開始タイムスタンプを記憶する段階と、第１の時間よりも遅い第２の時間において制御ノードから第２の命令を送信する段階を含む。各他のノードは、開始タイムスタンプ値と現在タイムスタンプ値との間の差を計算し、計算された差を現在タイムスタンプ値に加算する。他の実施例では、制御ノードはプリセット命令を送信し、各他のノードは上記プリセット命令に応じて夫々のタイムスタンプの値を指示された初期タイムスタンプ値にプリセットする。各他のノードは現在のタイムスタンプカウンタ値を指示されたリセット値と周期的に比較し、夫々の現在のタイムスタンプカウンタ値と指示されるリセット値との間の一致が検出されると、次のプリセット値に応じて次のタイムスタンプ値補正が行われるべきであることを示す同期フラグラッチを設定する。

【００２８】

本発明はまた上述の方法を実施するネットワークに関する。

本発明の上述及び他の特徴、目的、及び利点は、以下の説明を添付の図面と共に読むことによってより明らかに理解されよう。

【００２９】

ここで図４を参照するに、無線ＡＴＭネットワーク内のフレーム同期のために本発明の典型的な実施において使用される媒体アクセス制御（ＭＡＣ）サブシステム５１のハードウエアブロック図が示されている。概して、このハードウエアは、ＭＡＣプロトコルにおける変形及び改善を入れるためにかなりプログラム可能である。これは、ＭＡＣ層のスケジューリング及び管理機能が最小量の遅延又はパケット損失で実行されることを可能とするためＡＴＭ及び物理（ＰＨＹ）層の間にバッファリングされたデータ路を与える。

【００３０】

ダウンロードデータ路は、第２のパイプへのスイッチが必要であればパケット損失なしにこれを実行することができ、同時にＵＴＯＰＩＡデータレートといった高いデータレートを順応させることができるようＡＴＭデータフローを緩和するための入力ＦＩＦＯ５３と、スケジューリングが行われる２つのＳＡＲＡＭ又は優先順位が付けられたバッファ５５と、ＰＨＹ層データレートを順応させるための出力ＦＩＦＯ５７とを含む。アップロードデータ路は、パケットを収集するＳＡＲＡＭ又はアップロードバッファ５９を含み、ＭＡＣ層データへのＲＡＭアクセスを可能とし、ＡＴＭ層への順次アクセスを続ける。メールボックス６１（例えばデュアルポートＲＡＭ即ちＤＰＲＡＭ）は、例えばパラメータ及びステータス情報の交換を可能とするＭＡＣ及びＡＴＭ層の間のメールボックス機能のために設けられる。一対のプログラム可能論理装置（ＰＬＤ）６２及び６３は、インターフェース、データ路、及びタイムキーピング機能を制御するために使用される。プロセッサ又はマイクロプロセッサユニット（ＭＰＵ）６５は、全てのスケジューリング及び管理機能を実行するために符号化される（プログラムされる）。

【００３１】

実際の典型的な実施では、２つのプログラム可能装置６２、６３は、ＡｌｔｅｒａＦＰＬＤであり、ＡＡｌｔｅｒａ＿１@及びＡＡｌｔｅｒａ＿２と称され、ＭＰＵ６５はＲ３０００クラスのＩＤＴ７９Ｒ３０４１　ＭＩＰＳ　ＲＩＳＣプロセッサである。Ａｌｔｅｒａ＿１は、プロセッサ対話用のメモリマップ、命令及びステータスレジスタを実施するアドレスラッチ及びチップ選択復号化器と、ＵＴＯＰＩＡを通じてＡＴＭ層とインタフェース接続されるよう設定される信号とを含む。Ａｌｔｅｒａ＿２はタイムスタンプカウンタ及び６つの比較器、即ちＣＤＦのフェーズに対する４つの比較器、ビーコン又はタイムスタンプの再複製間隔のための１つの比較器、及び関連づけのためのもう１つの比較器と共にプロセッサを補完する。更に、Ａｌｔｅｒａ＿２は、物理層インタフェース信号セット、及びプロセッサと対話するための命令及び状態レジスタを含む。

【００３２】

最大の柔軟性及び効率のため、無線ＡＴＭ（ＷＡＴＭ）ネットワーク内の基地局（ＢＳ）及び各無線端末（ＷＴ）の両方における使用のために共通のハードウエア設計が使用される。これは、必要に応じてハードウエアをＢＳ又はＷＴのいずれかの使用のために適合させるために、ＥＥＰＲＯＭ６９（又は他の適当なメモリ装置）の中に二組の操作コードを与え、二組の操作コードのうちの選択された方をパワーアップ時にＳＲＡＭ７１（又は他の適当なメモリ）の中へ呼び出すためにスイッチ選択又は他の適当な技術を用いることによって達成される。

【００３３】

実際の典型的な実施例では、ブートコード及び二組の操作コード（ＢＳ及びＷＴ）を保持するためにＡＭ２９Ｆ０１０１２８Ｋ　ＥＥＰＲＯＭ装置が用いられ、動的パラメータ及び一般的なワークスペースの記憶のためにＩＤＴ７１２５６１２８ＫＳＲＡＭが用いられる。スケジューリング及びＭＡＣ管理機能が実行される実際のバッファは、一つのポートを通じてランダムアクセスが行われ、他のポートを通じて順次アクセスが行われる、ＩＤＴ７０８２５デュアルポートＲＡＭ（ＤＰＲＡＭ）である。これらは、順次ポートを通じて入力されるデータがランダムアクセスポートを通じてＭＰＵによって操作され、やはり順次ポートを通じて次の層へ渡されることを可能とする。

【００３４】

概して、基本的な概念は、タイムスタンプ情報を、タイムスタンプ値の伝送の正確な決定論的なスケジューリングを（予め）可能とするよう送信し、所与の（固定の）タイムスロット中にタイムスタンプ値のみを送信する必要性を除去することである。

【００３５】

本発明の第１の実施例によれば、「ｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔ（タイムスタンププリセット）」命令はＢＳによってネットワーク内の全てのＷＴへ送信される。ｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔ命令を受信すると、全てのＷＴはそれらのタイムスタンプをゼロ（又は物理層によって生ずる遅延に等しい値）にプリセットする。これは（第二の実施例に対して）１つのステップを除去し、大局的な時間情報の損失をもたらす。

【００３６】

本発明の第二の実施例によれば、タイムスタンプ情報は２つのステップで送信される。まず、ＢＳによって「ｔｉｍｅｓｔａｍｐ＿ｇｅｔ（タイムスタンプ獲得）」命令が送信され、これに応じて現在のタイムスタンプ値はＢＳを含む全ての端末においてレジスタの中に記憶される。次に、後の時間に、「ｔｉｍｅｓｔａｍｐ＿ｌｏａｄ（タイムスタンプロード）」命令が送信され、これは最後のｔｉｍｅｓｔａｍｐ＿ｇｅｔ命令の間にＢＳレジスタ内に記憶されたタイムスタンプ値を送信する。次に、受信器はＢＳレジスタの中に記憶されたタイムスタンプ値とそれらの受信器の中の記憶されたタイムスタンプ値との差を計算し、この差をそれらの受信器の現在のタイムスタンプ値に加える。

【００３７】

第１の実施例の基本的な手順は以下の通りである。ＢＳ及び全てのＷＴは夫々、補正が行われるであろうことを示す同期フラグラッチをいつ設定するかを決定するために、それらのタイムスタンプカウンタの現在値を（全ての端末に対して同じである）固定値と比較する。例えば、値は、補正が命令される前に１０μｓ毎にネットワーク内で全てのタイムスタンプ間に１μｓのスキューが生ずることを可能とするよう選択されうる。全てのプロセッサは、ＣＤＦのダウンリンクフ

ェーズ中にこれらの同期フラグについてポーリングし、設定されていれば、このイベントを同時に検出する。すると次のＣＤＦのＢＳ＿ＳＩＧフェーズ中にタイムスタンプ値補正が行われる。タイムスタンプ値補正は、単に全てのＷＴに対する適当なヘッダバイトでありうるｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔ命令を送信するＢＳによって開始される。

【００３８】

全てのＷＴ及びＢＳは、全ての端末が時間的にできる限り近くヘッダバイトに対して作用することを確実にするためにそれらのＰＨＹ層インタフェースにおいてこのヘッダバイトを探す。それらの間の遅延は、一定である物理路の遅延のみである。タイムスタンプ値補正は、プロセス全体が決定論的であるよう、プロセッサの介在無しに完全にハードウエアによって行われることが望ましい。

【００３９】

ここで図５を参照するに、ＢＳのｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔブロック８０及びＰＨＹ層インタフェース８２が示されている。ＰＨＹ層インタフェース８２は、指定されたＢＳ＿ＳＩＧフェーズが開始したときにｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔバイトを送信されるべき最初のバイトとして保持するＰＨＹ　ＦＩＦＯ８４を含む。ｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔバイトは、ｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔブロック８０の中でＰＨＹ層８６及びバイトレジスタ８８の両方へ同時に送信される。比較器８９は、レジスタ８８によって捕捉されたｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔ値を固定のタイムスタンプ（ＴＳ）ヘッダ値９０と比較し、一致が検出されると、状態機械９２を始動させる。

【００４０】

状態機械９２は、始動された後、タイムスタンプカウンタ９４の中に記憶された現在のタイムスタンプ値を読み出し、読み出された現在のタイムスタンプ値をタイムスタンプレジスタ９６の中へロードする。状態機械９２は次に、タイムスタンプカウンタ９４を物理層を通じた遅延に対応する一定のオフセット（プリセット）値９５でプリセットする（ＢＳでは、ゼロからこの遅延を差し引く）。次に、状態機械９２は、続くデータに対する誤った比較を防止するために、バイトレジスタ比較器８９を使用不可とする。比較器９８は、タイムスタンプカウンタ

９４によって出力される現在のタイムスタンプ値を固定値１００と比較し、一致が検出されると、補正が行われるであろうことを示す同期フラグラッチ１０２を設定する。上述のように、この固定値は、補正が命令される前に１０μｓ毎にネットワーク内で全てのタイムスタンプ間に１μｓのスキューが生ずることを可能とするよう選択されうる。

【００４１】

ここで図６を参照するに、ＷＴのｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔブロック１１０及びＰＨＹ層インタフェース１１２が示される。ＷＴのｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔブロック１１０はＢＳのｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔブロックと同一であり、従って、二つのブロック中の同様の要素は同じ参照番号で示され、但し、ｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔブロック１１０中の要素を示す参照番号にはプライム（’）が付されている。また、各ＷＴのハードウエアは同一となることが認められよう。ＷＴにおいて、ヘッダバイトはパケットの最初のバイトとして受信される。これは同時にバイトレジスタ８８’及びアップロードバッファ１１２の中に記憶される。比較器８９’は、バイトレジスタ８８’によって捕捉されたｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔ値を固定タイムスタンプ（ＴＳ）ヘッダ値９０’と比較し、一致が検出されると、状態機械９２’を始動させる。

【００４２】

状態機械９２’は、始動された後、タイムスタンプカウンタ９４’の中に記憶された現在のタイムスタンプ値を読み出し、読み出された現在のタイムスタンプ値をタイムスタンプレジスタ９６’の中へロードする。状態機械９２’は次に、タイムスタンプカウンタ９４’を物理層を通じた遅延に対応する一定のオフセット（プリセット）値９５’でプリセットする。次に、状態機械９２’は、続くデータに対する誤った比較を防止するために、バイトレジスタ比較器８９’を使用不可とする。比較器９８’は、タイムスタンプカウンタ９４’によって出力される現在のタイムスタンプ値を固定値１００’と比較し、一致が検出されると補正が行われるであろうことを示す同期フラグラッチ１０２’を設定する。

【００４３】

タイムスタンプ値補正プロセスは両端で同一のハードウエアによって行われるため、近い同期が達成される。ＢＳと各ＷＴとの間の遅延の量は、単に信号の伝搬遅延である。

【００４４】

保持レジスタ９６及び９６’の中に記憶されるタイムスタンプ値は、必要に応じて同一送受信器内でＭＡＣ層とインタフェース接続されうる他のプロトコル層に対するする補正を計算するために使用されうる。この動作を容易とするため、ＢＳはそのＴＳレジスタ値を典型的には次のＧＳ＿ＳＩＧフェーズにおいて（Ａ／Ｄバスを通じて）別個のメッセージの中で送信せねばならない。すると、各ＷＴは、ＢＳのＴＳレジスタ値とそのＷＴのＴＳレジスタ値との間の差を決定し、従ってＢＳからの時間オフセットの量を決定する。他のプロトコル層はこのオフセットを所望であればそれらのタイムスタンプ値を補正するために使用しうる。従って、この場合、ＭＡＣ層は大局的な時間基準を失うが、それでもなおフレーム同期を達成することが可能である。

【００４５】

ここで図７を参照するに、本発明の第二の実施例によって設計され、ＢＳ及びＷＴの両方に共通なｔｉｍｅｓｔａｍｐ＿ｌｏａｄブロック１２１のブロック図が示されている。容易にわかるように、ｔｉｍｅｓｔａｍｐ＿ｌｏａｄブロック１２１は図５及び６に夫々示されるｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔブロック８０及び１１０の簡単な変更を構成する。従って、同様の要素は同じ参照番号で示され、但し、ｔｉｍｅｓｔａｍｐ＿ｌｏａｄブロック１２１の要素を示す参照番号にはダブルプライム（”）が付されている。

【００４６】

以下明らかとなるように、本実施例では、大局的な時間基準はＭＡＣ層の中に維持されうる。この実施例では、タイムスタンプ情報は２つのステップで送信される。まず、上述のｔｉｍｅｓｔａｍｐ＿ｐｒｅｓｅｔ命令と同様の「ｔｉｍｅｓｔａｍｐ＿ｇｅｔ」命令が送信される。この命令はパラメータを有さず、１バイト中で符号化されうる。ＢＳ及び全てのＷＴは、このｔｉｍｅｓｔａｍｐ＿ｇｅｔ命令が検出されたときに、夫々のタイムスタンプ（ＴＳ）レジスタ９６”を

ロードする。第二に、ＢＳは、最後のｔｉｍｅｓｔａｍｐ＿ｇｅｔ命令が送信されたときにそのＴＳレジスタ９６”の中に保持されるタイムスタンプ値をパラメータとする「ｔｉｍｅｓｔａｍｐ＿ｌｏａｄ」を発行する。各ＷＴはすると、受信されたＢＳのタイムスタンプ値とそのＷＴのＴＳレジスタ９６”の中に保持されたタイムスタンプ値との間の差を計算し、次にこの差（オフセット）値をオフセットレジスタ１２３の中に記憶する。次に、ＷＴにおけるプロセッサはＡＬＯＡＤ@命令を送信し、これは加算器１２５にオフセットレジスタ１２３からのオフセット値をタイムスタンプカウンタ９４”の現在値へ加算させ、次に和をタイムスタンプカウンタ９４”の中へ再びロードさせる。この操作は時間によって重大な影響を受けるものではないことが認識されるべきである。

【００４７】

この実施例は、最大で３２ビットでありうる加算器１２５の追加的な複雑さを導入する。しかしながら、この操作は時間によって重大な影響を受けないため、３２ビット全加算器を実施する必要はない。例えば、多数のクロックサイクルに亘って加算を行うことにより４ビット加算器を使用し、この時間期間中はタイムスタンプカウンタ９４”を使用不可とすることが可能である。このクロックサイクルの損失は、プロセッサソフトウエアによってオフセットレジスタ１２３の中で順応されるべきである。

【００４８】

Ｔｉｍｅｓｔａｍｐ＿ｌｏａｄ命令が送信される前に、多数のｔｉｍｅｓｔａｍｐ＿ｇｅｔ命令が送信されうることが可能である。この場合、ＢＳは各ｔｉｍｅｓｔａｍｐ＿ｇｅｔ命令の後にＴＳレジスタ９６”の値を複製し、全ての係る値の合計と等しいパラメータと共にｔｉｍｅｓｔａｍｐ＿ｌｏａｄ命令を送信しうる。ＷＴは各ｔｉｍｅｓｔａｍｐ＿ｇｅｔ命令の後にＴＳレジスタ９６”の中の全ての値の和を計算し、次にｔｉｍｅｓｔａｍｐ＿ｌｏａｄ命令を通じて受信されたパラメータ値とこの計算された和との差を計算し、オフセットレジスタ１２３をこの差の値で設定する。

【００４９】

タイムスタンプ値の伝送を２つのステップへ分けることにより、データ及びタ

イムスタンプパケットは実際の伝送時間の前に伝送のためにスケジューリングされえ、従ってＭＡＣプロトコルのためのソフトウエアの設計における大きな柔軟性を可能とする。

【００５０】

本発明は上述において詳細に説明されたが、当業者によって明らかとなるであろう本発明の概念の多くの変形及び／又は変更は添付の請求項に記載される本発明の精神及び範囲を逸脱しないことが理解されるべきである。

【００５１】

例えば、ｔｉｍｅｓｔａｍｐ＿ｇｅｔ及びｔｉｍｅｓｔａｍｐ＿ｌｏａｄ命令の両方はＢＳといった同一の端末（ノード）によって始動される必要は無いことが理解されるべきである。むしろ、従来技術より公知であるように、ネットワーク内の全てのノードがネットワーク制御ノード又はＢＳとして作用するよう任意の時間において動的に再割当てされうるネットワークを使用することが可能である。かかるネットワークは従って、分散された同期の可能性を与える。従って、現在は制御ノードとして割り当てられているノードは当該の命令、例えばｔｉｍｅｓｔａｍｐ＿ｇｅｔ又はｔｉｍｅｓｔａｍｐ＿ｌｏａｄ命令を発生するために使用されうる。

【図面の簡単な説明】

【図１】

ＩＥＥＥ１３９４シリアルバスのサイクルマスターのｃｙｃｌｅ＿ｓｔａｒｔパケット発生部を示すブロック図である。

【図２】

ＩＥＥＥ１３９４シリアルバスに接続された受信器ノードのｂｕｓ＿ｔｉｍｅリセット部を示すブロック図である。

【図３】

無線ＡＴＭネットワーク用の予約ベースＭＡＣプロトコルにおいて使用される周期的制御データフレーム（ＣＤＦ）の構造を示す図である。

【図４】

本発明の典型的な実施例において使用されるＭＡＣサブシステムを示すブロッ

ク図である。

【図５】

本発明の第１の実施例によって実施される基地局のタイムスタンププリセットブロックを示すブロック図である。

【図６】

本発明の第１の実施例によって実施される無線端末のタイムスタンププリセットブロックを示すブロック図である。

【図７】

本発明の第２の実施例によって実施される基地局及び無線端末のタイムスタンプロードブロックを示すブロック図である。

【図１】

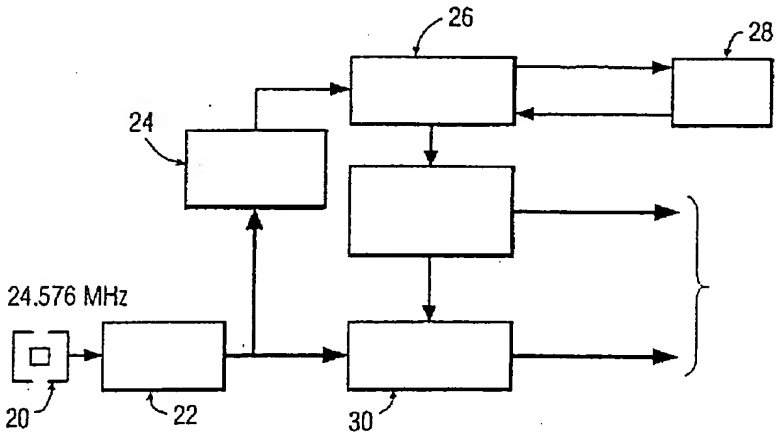


FIG. 1

【図２】

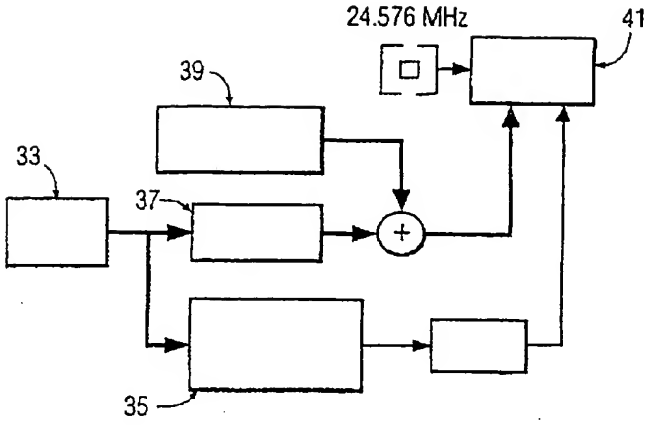
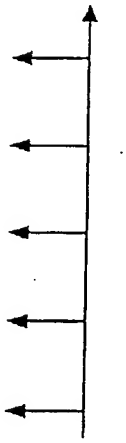


FIG. 2

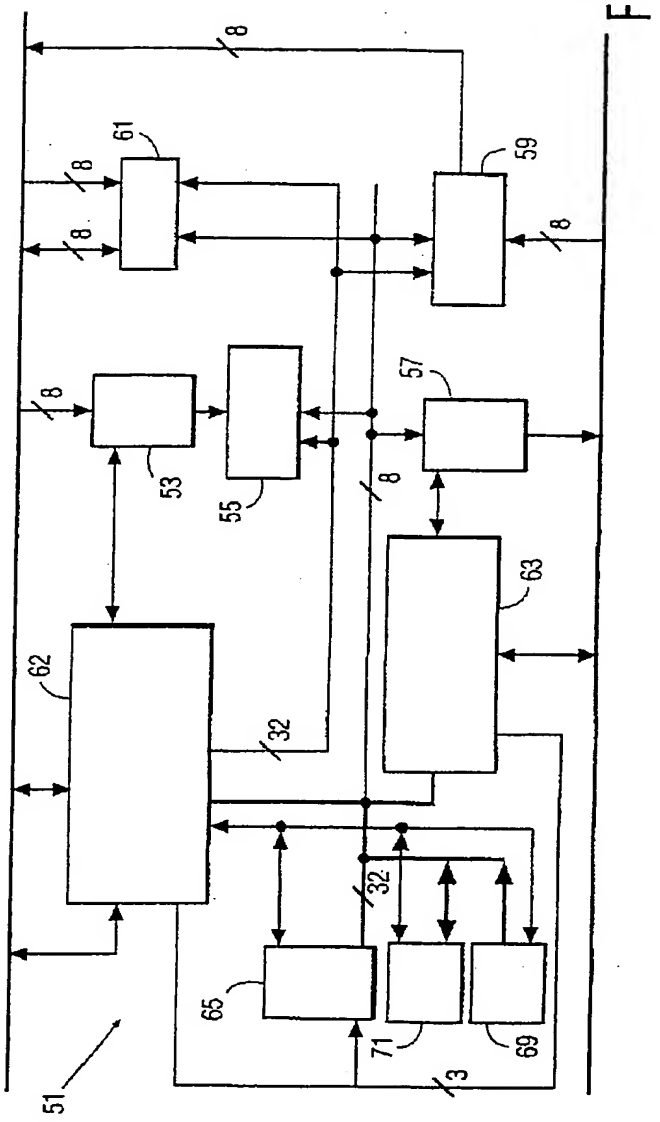
【図３】

FIG. 3



【図４】

FIG. 4



【図５】

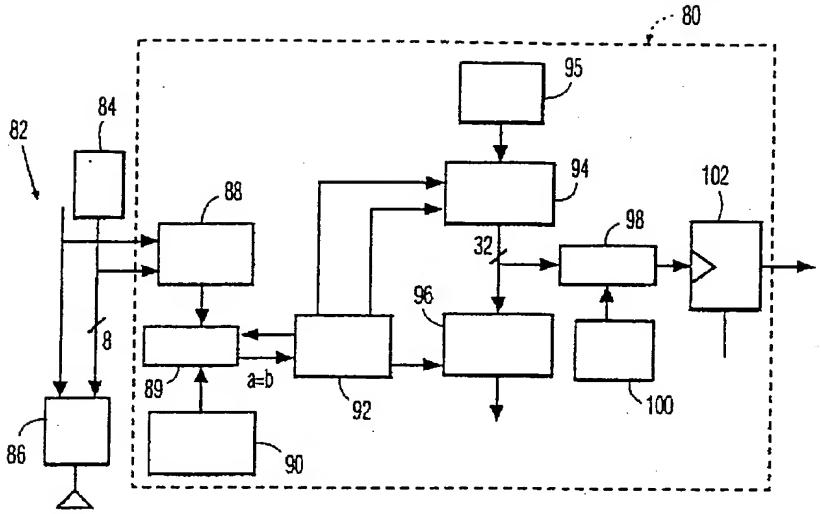


FIG. 5

【図６】

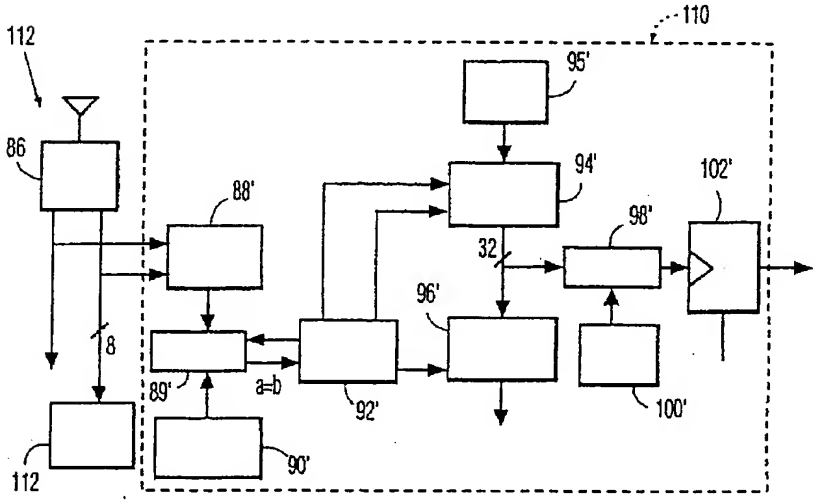


FIG. 6

【図７】

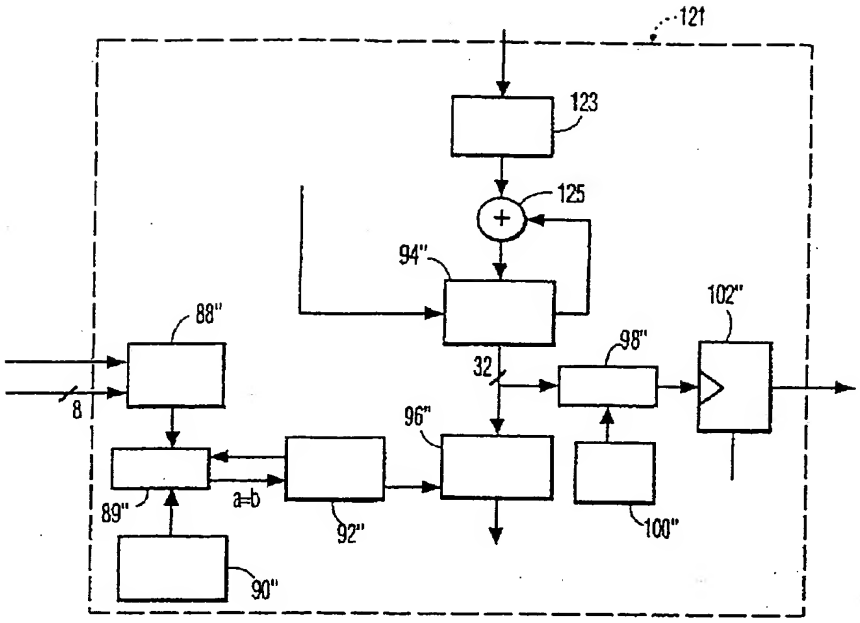


FIG. 7

【国際調査報告】

1

INTERNATIONAL SEARCH REPORT

International application No.
PCT/IB 99/00928

A. CLASSIFICATION OF SUBJECT MATTER

IPC7: H04J 3/06, H04L 7/00
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC7: H04J, H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

SE,DK,FI,NO classes as above

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5602992 A (G. DANNEELS), 11 February 1997 (11.02.97), column 7, line 38 - column 8, line 45 --	1-17
A	US 5598352 A (M.A. ROSENAU ET AL), 28 January 1997 (28.01.97), column 11, line 43 - column 12, line 45 --	1,3-10, 12-15,17
A	US 5012469 A (K. SARDANA), 30 April 1991 (30.04.91), column 2, line 55 - column 3, line 42, figure 1, abstract --	2,11,16

[X] Further documents are listed in the continuation of Box C. [X] See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" erlier document but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
14 December 1999	07 -01- 2000
Name and mailing address of the ISA/ Swedish Patent Office Box 5055, S-102 42 STOCKHOLM Facsimile No. +46 8 666 02 86	Authorized officer Per Källquist/cs Telephone No. +46 8 782 25 00

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/IB 99/00928

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5550873 A1 (D. DOLEV ET AL), 27 August 1996 (27.08.96), column 1, line 40 - line 50; column 8, line 54 - column 9, line 6, abstract	1-17

--

INTERNATIONAL SEARCH REPORT
Information on patent family members

02/12/99

International application No.
PCT/IB 99/00928

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5602992 A	11/02/97	GB 2284327 A,B	31/05/95
		GB 9416276 D	00/00/00
		SG 47984 A	17/04/98
US 5598352 A	28/01/97	EP 0797893 A	01/10/97
		JP 10511513 T	04/11/98
		WO 9619078 A	20/06/96
		EP 0783824 A	16/07/97
		JP 10507597 T	21/07/98
		US 5594660 A	14/01/97
		US 5815634 A	29/09/98
		US 5838380 A	17/11/98
		US 5923665 A	13/07/99
		WO 9610889 A	11/04/96
US 5012469 A	30/04/91	NONE	
US 5550873 A1	27/08/96	US 5784421 A	21/07/98
		JP 2069087 C	10/07/96
		JP 6216890 A	05/08/94
		JP 7105787 B	13/11/95
		US 5428645 A	27/06/95

フロントページの続き

(71)出願人 Ｇｒｏｅｎｅｗｏｕｄｓｅｗｅｇ １，
5621 ＢＡ Ｅｉｎｄｈｏｖｅｎ，Ｔｈｅ Ｎｅｔｈｅｒｌａｎｄｓ
(72)発明者 スタイナー，レオン
オランダ国，5656 アーアー アインドーフェン，ブロフ・ホルストラーン ６

Ｆターム(参考) 5K028 HH02 HH03 MM08 MM12 NN01
NN31 NN57 SS24 SS28
5K047 AA18 CC06 GG11 GG16 HH43
HH55 MM24 MM28 MM56

【要約の続き】
正が行われるべきであることを示す同期フラグラッチを設定する。上述の方法を実施するネットワークもまた開示される。

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04L 7/00

A2

(11) International Publication Number: WO 99/62216

(43) International Publication Date: 2 December 1999 (02.12.99)

(21) International Application Number: PCT/IB99/00928

(22) International Filing Date: 20 May 1999 (20.05.99)

(30) Priority Data:
09/086,270 28 May 1998 (28.05.98) US

(71) Applicant: KONINKLIJKE PHILIPS ELECTRONICS N.V. [NL/NL]; Groenewoudseweg 1, NL–5621 BA Eindhoven (NL).

(71) Applicant *(for SE only)*: PHILIPS AB [SE/SE]; Kottbygatan 7, Kista, S–164 85 Stockholm (SE).

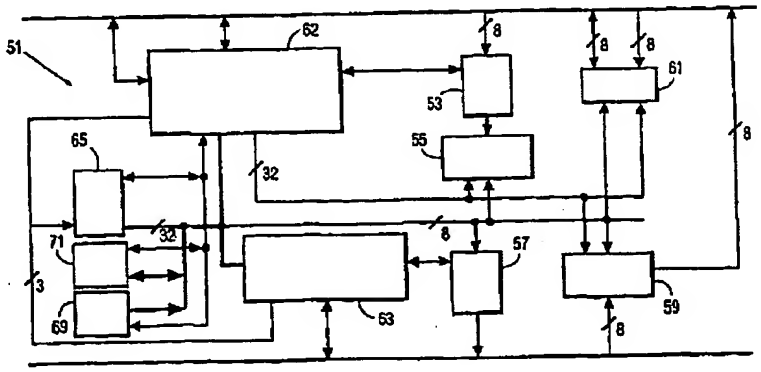
(72) Inventors: HULYALKAR, Samir, N.; Prof. Holstlaan 6, NL–5656 AA Eindhoven (NL). SHEKHTER, Elik; Prof. Holstlaan 6, NL–5656 AA Eindhoven (NL). STEINER, Leon; Prof. Holstlaan 6, NL–5656 AA Eindhoven (NL).

(74) Agent: GROENENDAAL, Antonius, W., M.; Prof. Holstlaan 6, NL–5656 AA Eindhoven (NL).

(81) Designated States: JP, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: METHOD OF TIMESTAMP SYNCHRONIZATION OF A RESERVATION–BASED TDMA PROTOCOL



(57) Abstract

A method for synchronizing timestamps in a network (e.g., a wireless ATM network) that includes a control node and a plurality of other nodes that communicate with one another over a common channel mediated by a medium–access control subsystem (e.g., one that uses a reservation–based TDMA protocol). The method includes the steps of sending a first command from the control node, at a first time, then storing a starting timestamp value in a register within the control node and each of the other nodes, in response to the first command, and then sending a second command from the control node, at a second time later than the first time. Each of the other nodes computes the difference between the starting timestamp value and a current timestamp value, and than adds the computed difference to the current timestamp value. In an alternative embodiment, the control node sends a preset command and each of the other nodes presets their respective timestamp to a prescribed initial timestamp value, in response to the preset command. Each of the nodes periodically compares a current timestamp counter value with a prescribed reset value, and upon detecting a coincidence, sets a sync flag latch to indicate that a next timestamp value correction is to be made, in response to a next preset command. Networks that implement the above–described methods are also disclosed.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Paso	GR	Greece	ML	Mali	TR	Turkey
BG	Bulgaria	HU	Hungary	MN	Mongolia	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MR	Mauritania	UA	Ukraine
BR	Brazil	IL	Israel	MW	Malawi	UG	Uganda
BY	Belarus	IS	Iceland	MX	Mexico	US	United States of America
CA	Canada	IT	Italy	NE	Niger	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NL	Netherlands	VN	Viet Nam
CG	Congo	KE	Kenya	NO	Norway	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NZ	New Zealand	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	PL	Poland		
CM	Cameroon	KR	Republic of Korea	PT	Portugal		
CN	China	KZ	Kazakstan	RO	Romania		
CU	Cuba	LC	Saint Lucia	RU	Russian Federation		
CZ	Czech Republic	LI	Liechtenstein	SD	Sudan		
DE	Germany	LK	Sri Lanka	SE	Sweden		
DK	Denmark	LR	Liberia	SG	Singapore		
EE	Estonia						

Method of timestamp synchronization of a reservation-based TDMA protocol.

BACKGROUND OF THE INVENTION

The present invention relates generally to systems and methods that enable multiple-access of the same channel in a network, and, more particularly, to a system and method of timestamp synchronization in a network that employs a reservation-based TDMA protocol.

In general, communications networks, particularly wireless networks, typically employ a multiple-access protocol that is designed to prevent collisions of data packets due to simultaneous transmission of the data packets by multiple transmitters in the network using the same channel. One protocol that has come into widespread use is known as Time-Division Multiple Access (TDMA). A detailed description of this technique can be found in the reference book *Telecommunications Networks: Protocols, Modeling and Analysis*, Addison-Wesley, 1997. In general, in accordance with the TDMA protocol, channel time is divided into small time slots, each of which is assigned to a different node (user). This time slot assignment can either be fixed (classical TDMA), or variable (reservation-based TDMA). In either case, since the number of nodes (users) is finite the data is usually transmitted in TDMA "frames", which ensure that the delays encountered by the different users are finite.

For example, in fixed-assignment TDMA, the TDMA frame consists of the total number of slots assigned to all users, after which the TDMA frame repeats. In the case of reservation-based TDMA, a natural framing occurs in terms of different "phases" of the TDMA frame, consisting typically of a "control" phase in which reservations are requested and assigned, and a "data" phase in which the data is transmitted by the different users in their respective assigned time slots.

It is necessary that all transmitters and receivers in the TDMA network be synchronized in terms of the TDMA frame. An incorrectly synchronized transceiver, at best, cannot communicate, but, at worst, can cause the entire TDMA network to collapse if appropriate safeguards are not built into the protocol. It should be recognized that TDMA frame synchronization is not the same as clock synchronization of a modem, which is a function of the Physical layer. Usually, frame synchronization is achieved using a centralized

control strategy implemented by a central controller (CC). However, frame synchronization can also be implemented in a distributed fashion.

In most TDMA networks, a universal time reference is required to properly allocate resources for transmission. This universal time reference is usually provided in the form of a "timestamp", e.g., which specifies the current time, such as 2 p.m. The timestamps are broadcast periodically by the central controller, and are used by the end terminals (ETs) to synchronize their "timestamp" registers.

For variable-sized TDMA frames, synchronization achieved through the use of timestamps typically requires the utilization of a phase-locked loop (PLL) in each of the ETs, which can be quite complex. Further, the PLLs used for this purpose must be redesigned whenever the parameters of the timestamp scheme are changed, for example, when the frequency of timestamp transmission is changed. In this connection, a generic synchronization scheme is desired in order to enable an ET to be used interchangeably within many different networks.

With presently known frame synchronization techniques, the timestamp values are generated from a common clock. For example, in accordance with the IEEE 1394 standard, the clock distribution within a local 1394 serial bus is accomplished by means of the cycle master (or the central controller) periodically sending a cycle_start packet , which contains the current bus_time (or timestamp). This "re-synchronization" is ideally performed every 125 ms. However, it is possible that the transmission of the cycle_start packet could be slightly delayed due to the local 1394 serial bus being used by another node when the cycle_start packet needs to be sent, thereby requiring the cycle master to wait until that node finishes its transmission before sending the cycle_start packet.

With reference now to FIG. 1, a method for generating the cycle_start packet at the 1394 cycle master will now be described. More particularly, a crystal 20 that runs at the master clock rate of 24.576 MHz is input to a cycle counter 22. Frame synchronization is achieved by synchronizing the cycle counters within all 1394 nodes interconnected by the local 1394 serial bus. The output of the cycle counter 22 is passed through a modulo 125 ms block 24 that sends a trigger signal to a state machine 26 every 125 ms, in response thereto. The state machine 26 then sends a channel request signal to the 1394 Physical (PHY) layer 28 in response to the trigger signal sent by the modulo 125 ms block 24. As soon as the channel becomes available, the 1394 PHY layer 28 sends back a channel available signal to the state machine 26. In response to the channel available signal, the state machine 26 prepares the packet header for the cycle_start packet, and sends an enable signal to a register 30 to latch the

contents of the cycle counter 22 at the proper instant to generate the current bus_time (timestamp value). Any delay in processing can be easily taken into account by properly delaying the enable signal to the register 30 by the amount of the processing delay.

At the receiver (i.e., each node that receives the cycle_start packet), the cycle counter in that node must be set to the appropriate bus_time received via the cycle_start packet. A method for accomplishing this task will now be described with reference to FIG. 2. More particularly, the PHY layer 33 of the receiver node receives the cycle_start packet and sends it to the link layer, which includes a decode cycle_start header block 35 for decoding the cycle_start packet header to ensure that it is indeed the cycle_start packet. Simultaneously, the bus_time value is extracted from the cycle_start packet and loaded into a register 37. A processing delay block 39 adds any processing delay (for the decoding operation and/or for the loading of the bus_time value into the register 37) to either the output of the register 37, or to a load signal output by the decode cycle_start header block 35. The load signal is applied to the load terminal of the cycle counter 41 of that receiver node to enable the loading of the sum of the contents of the register 37 and the processing delay into the cycle counter 41.

It should be recognized that it is extremely important to be cycle-accurate while resetting the cycle counter 41, which means that the bus_time transmitted by the cycle master must correspond to the exact time that the cycle_start packet is sent, and that the processing delay in each receiver node must be very precisely determined.

The resetting of the cycle counter 41 every 125 ms ensures that the clocks obtained from different crystals do not drift significantly with respect to each other. Most protocols have an interval during which the timestamp update must be sent. Otherwise, the timing jitter may be larger than what can be handled by a particular application, e.g., an MPEG decoder.

IEEE 802.11 has a similar protocol. More particularly, in accordance with the IEEE 802.11 standard, "beacons" are periodically broadcast, in a manner similar to the broadcast of the cycle_start packet in accordance with the IEEE 1394 standard. The beacon contains, among other fields, the timestamp value and the timestamp interval. The timestamp interval is flexible with the IEEE 802.11 standard, as contrasted to the fixed 125 ms cycle time with the IEEE 1394 standard. As with the IEEE 1394 standard, the beacon transmission can be delayed if the channel is not immediately available. As with the IEEE 1394 standard, the key requirement is that the timestamp value must correspond to the exact time of transmission of the beacon packet.

It should be recognized that both IEEE 1394 and IEEE 802.11 are not reservation-based protocols. In both cases, an arbitration phase is first initiated to determine access for a particular receiver node. Even the cycle_start and beacon packets must first arbitrate the use of the channel prior to transmission.

For a reservation-based TDMA protocol, there are many problems with the timestamp-based approach. The first problem is that the transmission of the timestamp value must also be reserved, and subsequently, other data must also be queued for transmission. In order to ensure efficient use of processor resources (which must be used for managing many other functions), this queuing is usually scheduled in advance. However, the timestamp value cannot be obtained until the exact time of transmission. Further, the queuing of the data packets behind the timestamp value cannot be done before the timestamp value is obtained. Of course, it is possible to switch the data stream between two separate queues with one holding the timestamp value and the other holding the data. However, this solution is quite complicated and requires precise synchronization.

A more detailed understanding of this problem can be gained by considering the case of a wireless asynchronous transfer mode (ATM) network that uses a reservation-based medium-access control (MAC) protocol. The MAC protocol implementation depends on a periodic control-data-frame (CDF), as described by C.Y. Ngo and S. N. Hulyalkar in ADetailed MAC Sub-System Description for BS-Oriented WATM LAN@, WP3, D3.1, Aug. 15, 1996. Each CDF contains many phases, during which both control and data information is sent from both the base station (BS) and the wireless terminal (WT). FIG. 3 illustrates four phases for implementing such a generic structure, namely, BS_SIG; DN_DATA; UP_DATA; and, E_BURST. A brief description of each of these phases follows:

BS_SIG: During this phase, the BS sends control information for the downlink. The timestamp packet can be assumed to be sent during this phase. At the BS, the processor starts the transmission of packets from BS. At the WT, the WT starts the process of reception of packets from the BS.

DN_DATA: During this phase, the BS sends data packets for the WTs. At the BS, the processor is busy interpreting the packets sent by the WT during the UP_DATA phase. At the WT, the processor is busy storing the PHY FIFO for the next burst of transmission during the UP_DATA phase.

UP_DATA: During this phase, the WT sends data and signaling packets for the BS. Signaling is sent using superslots. At the BS, the processor is busy storing the PHY FIFO for the next burst of transmission during the BS_SIG and DN_DATA phases. At the

WT, the processor is busy interpreting the packets sent by the BS during the BS_SIG and the DN_DATA phases.

E_BURST: During this phase, the WTs, which have not currently been allocated space for transmission during the UP_DATA phase, indicate whether they want to enter the WATM network. Both the WT and the BS processors are busy implementing the E_BURST phase.

The hardware design is based on the BS and each WT keeping the same timestamp values as a basis for computing the four phases of a CDF. All must maintain the same time periods in order to communicate and transfer packets effectively. All must synchronize their timestamps periodically, by copying the base station value, and all must take starting time directives from the BS.

The MAC processor is assumed to be interrupt-driven for both the WTs and the BS. The BS determines the timing for the entire system. Using the timestamp value as a reference, it determines the exact time when each of the phases operates. This timing information is sent during the BS_SIG_n phase. Since all phases are successive to each other, the WT and the BS set up a counter for the next phase based on the timing information, which then triggers an interrupt to the processor when the counter overflows. The processor must finish its functions during the respective phase within the time allotted and be prepared for the next phase.

For timestamp synchronization, the BS can be assumed to send a timestamp value during the BS_SIG phase. However, note that the BS is busy storing the PHY_FIFO with the packets intended for transmission during the BS_SIG and DN_DATA phases. However, the timestamp value must be determined during the BS_SIG phase and cannot be obtained during the UP_DATA phase. Consequently, the normal transmission stream must be stopped to allow for the timestamp value to be loaded from the timestamp register during the time of transmission. This solution is not desirable since it conflicts with the direct data path.

It should be appreciated that the problem described above is not due to the particular protocol considered, but is generally due to the reservation-based nature of the protocol, whereby decisions on what is transmitted at particular times are made in advance of those times.

One solution to this problem is to send only a timestamp packet during the BS_SIG phase. But for wireless ATM, which uses a 53-byte cell, this is a waste of resources, since a timestamp value may be only 4 bytes. More generally, the use of a fixed slot for transmission of only timestamp information results in a waste of network resources. In this

connection, it should be recognized that this fixed slot is an intrinsic aspect of any reservation-based protocol, because without the "smallest" slotting, it is not possible to reserve any channel time. The problem arises due to the fact that the smallest time slot is still significantly greater than the time required to send only the timestamp value.

Based on the above and foregoing, it can be appreciated that there presently exists a need in the art for a method of timestamp synchronization in a reservation-based TDMA-protocol network that overcomes the above-described drawbacks and shortcomings of the presently available technology. The present invention fulfills this need in the art.

SUMMARY OF THE INVENTION

The present invention encompasses a method for synchronizing timestamps in a network (e.g., a wireless ATM network) that includes a control node and a plurality of other nodes that communicate with one another over a common channel mediated by a medium-access control subsystem (e.g., one that uses a reservation-based TDMA protocol). The method includes the steps of sending a first command from the control node, at a first time, then storing a starting timestamp value in a register within the control node and each of the other nodes, in response to the first command, and then sending a second command from the control node, at a second time later than the first time. Each of the other nodes computes the difference between the starting timestamp value and a current timestamp value, and then adds the computed difference to the current timestamp value. In an alternative embodiment, the control node sends a preset command and each of the other nodes presets their respective timestamp to a prescribed initial timestamp value, in response to the preset command. Each of the nodes periodically compares a current timestamp counter value with a prescribed reset value, and upon detecting a coincidence, sets a sync flag latch to indicate that a next timestamp value correction is to be made, in response to a next preset command.

The present invention also encompasses networks that implement the above-described methods.

BRIEF DESCRIPTION OF THE DRAWINGS

These and other features, objects, and advantages of the present invention will become more clearly understood from the following detailed description read in conjunction with the attached drawings, in which:

FIG. 1 is a block diagram of a cycle_start packet-generating portion of the cycle master of an IEEE 1394 serial bus;

FIG. 2 is a block diagram of a bus_time resetting portion of a receiver node connected to an IEEE 1394 serial bus;

FIG. 3 is a diagram that depicts the structure of a periodic control-data-frame (CDF) used in a reservation-based MAC protocol for a wireless ATM network;

FIG. 4 is a block diagram of a MAC subsystem employed in an exemplary implementation of the present invention;

FIG. 5 is a block diagram of a timestamp-preset block of a base station implemented in accordance with a first embodiment of the present invention;

FIG. 6 is a block diagram of a timestamp preset block of a wireless terminal implemented in accordance with the first embodiment of the present invention; and,

FIG. 7 is a block diagram of a timestamp load block of a base station and wireless terminal in accordance with a second embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

With reference now to FIG. 4, there can be seen the hardware block diagram of a medium-access control (MAC) subsystem 51 that is employed in an exemplary implementation of the present invention for frame synchronization in a wireless ATM network. In general, this hardware is highly programmable in order to accommodate variations and improvements in the MAC protocol. It provides a buffered data path between the ATM and Physical (PHY) layers to allow for MAC layer scheduling and management functions to be executed with a minimum amount of delay or packet loss. The download data path includes an input FIFO 53 to cushion the ATM data flow so that if a switch to a second pipe becomes necessary it can be done without packet loss, while still accommodating high data rates, such as the UTOPIA data rate; two SARAMs or prioritized buffers 55 in which scheduling takes place; and, an output FIFO 57 to accommodate the PHY layer data rate. The upload data path includes an SARAM or upload buffer 59 that collects packets, allows RAM access to the MAC layer data, and continues sequential access to the ATM layer. A mailbox 61 (e.g., a dual-port RAM or DPRAM) is provided for mailbox functions between the MAC and ATM layers, e.g., for enabling the exchange of parameters and status information. A pair of programmable logic devices (PLDs) 62 and 63, are employed to control the interfaces, the

data paths, and the timekeeping functions. A processor or microprocessor unit (MPU) 65 is coded (programmed) to perform all scheduling and management functions.

In an actual exemplary implementation, the two programmable devices 62, 63, are Altera FPLDs, designated AAltera_1@ and AAltera_2, and the MPU 65 is an IDT79R3041 MIPS RISC processor of the R3000 class. Altera_1 contains the address latches and chip select decoders that implement the memory map, command and status registers for processor interaction, and the signal set to interface with the ATM layer via UTOPIA. Altera_2 complements the processor with the timestamp counter and six comparators assigned as follows: four to the phases of the CDF, one to the beacon or timestamp recopy interval, and one more for association. Additionally, Altera_2 contains the physical layer interface signal set, and the command and status registers to interact with the processor.

For maximum flexibility and efficiency, a common hardware design is utilized for use in both the base station (BS) and each of the wireless terminals (WTs) in the wireless ATM (WATM) network. This is accomplished by providing two sets of operating code in a EEPROM 69 (or other suitable memory device), and using a switch selection or other suitable technique to call a selected one of the two sets of operating code into an SRAM 71 (or other suitable memory) upon power-up, in order to configure the hardware for use in either a BS or WT, as appropriate.

In the actual exemplary implementation, an AM29F010 128 K EEPROM device is utilized to hold boot code and the two sets of operating code (BS and WT), and an IDT71256 128K SRAM is employed for storage of dynamic parameters and general work space. The actual buffers where scheduling and MAC management functions are performed are IDT70825 dual port RAMs (DPRAMs), with random access via one port and sequential access via the other port. They permit stream data to be entered via the sequential port to be operated on by the MPU via the random access port, and then passed onto the next layer, again via the sequential port.

In overview, the basic inventive concept is to send the timestamp information in such a manner as to permit precise, deterministic scheduling (in advance) of the transmission of the timestamp value and to eliminate the necessity of sending only the timestamp value during a given (fixed) time slot.

In accordance with a first embodiment of the present invention, a "timestamp_preset" command is sent by the BS to all WTs in the network. Upon receiving the timestamp_preset command, all WTs preset their timestamp value to zero (or equal to the

delay encountered through the physical layer). Although this eliminates a step (vis-à-vis the second embodiment), it results in the loss of the global time information.

In accordance with a second embodiment of the present invention, the timestamp information is sent in two steps. First, a "timestamp_get" command is sent by the BS, in response to which the current timestamp value is stored in a register at all terminals, including the BS. Then, at a later time, a "timestamp_load" command is sent, which sends the timestamp value stored in the BS register during the last timestamp_get command. Next, the receivers compute the difference between the timestamp value stored in the BS register and their stored timestamp value, and then add this difference to their current timestamp value.

The basic methodology of the first embodiment is as follows. The BS and all WTs each compare the current value of their timestamp counter against a fixed value (which is the same for all terminals) to determine when to set their sync flag latches, indicating that a correction will take place. For example, the value can be chosen to allow a 1 μs skew to develop between all timestamps in the network every 10 μs before a correction is commanded. All processors poll for these sync flags during the downlink phase of the CDF, and, if set, will detect this event concurrently. The timestamp value correction then takes place during the BS_SIG phase of the next CDF. The timestamp value correction is initiated by the BS sending a timestamp_preset command, which can simply be an appropriate header byte, to all of the WTs.

All of the WTs, and the BS, look for this header byte at their PHY layer interfaces in order to ensure that all terminals will act on the header byte as close in time as possible. The delay between them is the delay of the physical path only, which is constant. The timestamp value correction is preferably done entirely by hardware, without processor participation, so that the entire process is deterministic.

With particular reference now to FIG. 5, there can be seen a timestamp_preset block 80 and PHY layer interface 82 of the BS. The PHY layer interface 82 includes a PHY FIFO 84 that holds the timestamp_preset byte as the first byte to be transmitted when the designated BS_SIG phase begins. The timestamp_preset byte is concurrently transmitted to both the PHY layer 86 and a byte register 88 in the timestamp_preset block 80. A comparator 89 compares the timestamp_preset value captured by the byte register 88 with a fixed timestamp (TS) header value 90, and upon detecting a coincidence, initiates a state machine 92.

After being initiated, the state machine 92 reads the current timestamp value stored in the timestamp counter 94, and loads the read-out current timestamp value into a

timestamp register 96. The state machine 92 then presets the timestamp counter 94 with a fixed offset (preset) value 95 that corresponds to the delay through the physical layer (for the BS, it subtracts this delay from zero). Next, the state machine 92 disables the byte register comparator 89 in order to avoid erroneous comparisons on subsequent data. A comparator 98 compares the current timestamp value output by the timestamp counter 94 with a fixed value 100 and sets the sync flag latch 102 upon detecting a coincidence, indicating that a correction will take place. As previously mentioned, this fixed value can be chosen to allow a 1 μs skew to develop between all timestamps in the network every 10 μs before a timestamp value correction is commanded.

With reference now to FIG. 6, there can be seen a timestamp_preset block 110 and PHY layer interface 112 of a WT. It will be readily seen that the timestamp_preset block 110 of the WT is identical to the timestamp_preset block 80 of the BS, and, as such, like elements in the two blocks are indicated by the same numerals, except that the numerals designating the elements in the timestamp_preset block 110 are primed. It will also be appreciated that the hardware of each WT will be identical. At the WTs, the header byte is received as the first byte of a packet. It is concurrently stored in the byte register 88' and an upload buffer 112. A comparator 89' compares the timestamp_preset value captured by the byte register 88' with a fixed timestamp (TS) header value 90', and upon detecting a coincidence, initiates a state machine 92'.

After being initiated, the state machine 92' reads the current timestamp value stored in the timestamp counter 94', and loads the read-out current timestamp value into a timestamp register 96'. The state machine 92' then presets the timestamp counter 94' with a fixed offset (preset) value 95' that corresponds to the delay through the physical layer. Next, the state machine 92' disables the byte register comparator 89' in order to avoid erroneous comparisons on subsequent data. A comparator 98' compares the current timestamp value output by the timestamp counter 94' with the fixed value 100' and sets the sync flag latch 102' upon detecting a coincidence, indicating that a correction will take place.

Since the timestamp value correction process is performed by identical hardware at both ends, close synchronism is achieved. The amount of delay between the BS and each WT is merely the propagation delay of the signal.

Timestamp values stored in the holding registers 96 and 96' may be used as needed to compute corrections for other protocol layers that the MAC layer may interface within the same transceiver. To facilitate this operation, the BS must send its TS register value in a separate message (over an A/D bus), typically in the next BS_SIG phase. Each WT

then determines the difference between the BS TS register value and its TS register value and thus determines the amount of time offset from the BS. Other protocol layers can use this offset to correct their timestamp value if required. Thus, in this case, the MAC layer loses the global time reference, but is still able to achieve frame synchronization.

With reference now to FIG. 7, there can be seen a block diagram of a timestamp_load block 121 designed in accordance with the second embodiment of the present invention, and which is common to both the BS and the WTs. As will be readily seen, the timestamp_load block 121 constitutes a simple modification of the timestamp_preset blocks 80 and 110 depicted in FIGs. 5 and 6, respectively and described hereinabove. As such, like elements are indicated by the same numerals, except that the numerals designating the elements in the timestamp_load block 121 are double primed.

As will become apparent, with this embodiment, a global time reference can be maintained in the MAC layer. With this embodiment, the timestamp information is sent in two steps. First, a "timestamp_get" command is sent, which is similar to the timestamp_preset command discussed hereinabove. This command has no parameters and can be coded in one byte. The BS and all WTs load their respective timestamp (TS) register 96" when this timestamp_get command is detected. Second, the BS issues a "timestamp_load" command a parameter of which is the timestamp value held in its TS register 96" when the last timestamp_get command was sent. Each WT then calculates the difference between the received BS timestamp value and the timestamp value held in its TS register 96", and then stores this difference (offset) value in an Offset Register 123. Next, the processor at the WT sends a ALOAD@ command, which causes an adder 125 to add the offset value from the Offset Register 123 to the current value of the timestamp counter 94", and to then reload the sum into the timestamp counter 94". It should be appreciated that this operation is not time-critical.

This embodiment introduces the additional complexity of the adder 125, which can be as large as 32 bits. However, since this operation is not time-critical, it is not necessary to implement a full 32-bit adder. For example, it is possible to use a 4-bit adder by performing the addition over many clock cycles, with the timestamp counter 94" being disabled during this time period. This loss in clock cycles should be accommodated within the Offset Register 123 by the processor software.

It is possible that multiple timestamp_get commands can be sent before a timestamp_load command is sent. In this case, the BS can copy the value of the TS register 96" after every timestamp_get command and then send the timestamp_load command with the

parameter equal to the sum of all such values. The WTs can compute the sum of all the values in the TS register 96" after every timestamp_get command and then compute the difference between the parameter value received via the timestamp_load command and this computed sum, and then set the Offset Register 123 with this difference value.

By breaking the transmission of the timestamp values into two steps, data and timestamp packets can be scheduled for transmission before the actual time of transmission, thus allowing significant flexibility in the design of software for the MAC protocol.

Although the present invention has been described in detail hereinabove, it should be clearly understood that many variations and/or modifications of the basic inventive concepts taught herein which may appear to those skilled in the pertinent art will still fall within the spirit and scope of the present invention as defined in the appended claims.

For example, it should be appreciated that it is not necessary that both the timestamp_get and timestamp_load commands be initiated by the same terminal (node), such as the BS. Rather, as is generally known in the art, it is possible to utilize a network in which any node in the network can be dynamically re-assigned at any time to serve as the network control node or BS. Such networks thus provide the capability of distributed synchronization. Thus, the node which is presently assigned as the control node can be employed to generate the relevant command, e.g., the timestamp_get or timestamp_load command.

WHAT IS CLAIMED IS:

1. A method for synchronizing timestamps in a network that includes a control node and a plurality of other nodes that communicate with one another over a common channel mediated by a medium-access control subsystem, the method including the steps of:

sending a first command from the control node, at a first time;

storing a starting timestamp value in a register within the control node and each of the other nodes, in response to the first command;

sending a second command from the control node, at a second time later than the first time; and,

each of the other nodes computing a difference between the starting timestamp value and a current timestamp value, and then adding the computed difference to the current timestamp value.

2. The method as set forth in Claim 1, wherein the medium-access control subsystem uses a reservation-based TDMA protcol.

3. The method as set forth in Claim 1, wherein the first command comprises a timestamp_get command and the second command comprises a timestamp_load command.

4. The method as set forth in Claim 1, wherein any selected one of the nodes in the network can be assigned to serve as the control node at different times.

5. A method for synchronizing timestamps in a network that includes a control node and a plurality of other nodes that communicate with one another over a common channel mediated by a medium-access control subsystem, the method including the steps of:

sending a preset command from the control node;

each of the other nodes presetting a value of their respective timestamp to a prescribed initial timestamp value, in response to the preset command; and,

each of the other nodes periodically comparing a current timestamp counter value with a prescribed reset value, and upon detecting coincidence between their respective

current timestamp counter value and the prescribed reset value, setting a sync flag latch to indicate that a next timestamp value correction is to be made, in response to a next preset command.

6. The method as set forth in Claim 5, wherein the medium-access control subsystem uses a reservation-based TDMA protcol.

7 The method as set forth in Claim 5, wherein any selected one of the nodes in the network can be assigned to serve as the control node at different times.

8. A network, including:

a control node;

a plurality of other nodes;

a medium-access control subsytem (51) that mediates access to a communications channel used in common by the control node and the plurality of other nodes; and,

wherein the control node and the plurality of other nodes each include:

a comparator register (88, 88') that stores a preset command byte of a control packet transmitted by the control node at designated transmission times;

a first comparator (89, 89') that compares the value of the preset command byte with a prescribed header value and, upon detecting a coincidence, outputs a trigger signal;

a timestamp counter (94, 94');

a timestamp register (96, 96');

a sync flag latch (102, 102');

a state machine (92, 92') that, in response to the trigger signal, reads a timestamp value stored in the timestamp counter (94, 94'), loads the read-out timestamp value into the timestamp register (96, 96'), and presets the timestamp counter (94, 94') with a prescribed preset value; and,

a second comparator (98, 98') that compares the current timestamp value stored by the timestamp counter (94, 94') with a prescribed reset value and, upon detecting a coincidence, sets the sync flag latch (102, 102') to indicate that a next timestamp value correction is to be made, in response to a next preset command byte.

9. The network as set forth in Claim 8, wherein the prescribed preset value for the control node is zero.

10. The network as set forth in Claim 8 or 9, wherein the prescribed preset value for each of the other nodes corresponds to a delay through a physical layer.

11. The network as set forth in Claim 8, wherein the medium-access control subsystem (51) uses a reservation-based TDMA protcol.

12. The network as set forth in Claim 8, wherein the designated transmission times correspond to BS_SIG phases of control-data-frames.

13.The network as set forth in Claim 8, wherein any selected one of the nodes in the network can be assigned to serve as the control node at different times.

14. A network, including:

a control node;

a plurality of other nodes;

a medium-access control subsystem (51) that mediates access to a communications channel used in common by the control node and the plurality of other nodes; and,

wherein the control node and the plurality of other nodes each include:

a comparator register (88") that stores a command byte of a control packet transmitted by the control node at first times;

a first comparator (89") that compares the value of the command byte with a prescribed header value and, upon detecting a coincidence, outputs a trigger signal;

a timestamp counter (94");

a timestamp register (96");

a sync flag latch (102");

an offset register (123)";

a state machine (92") that, in response to the trigger signal, reads a timestamp value stored in the timestamp counter (94"), and loads the read-out timestamp value into the timestamp register (96"); and,

processor circuitry that computes the difference between a last timestamp value transmitted by the control node at second times later than the first times, and a current timestamp value stored in the timestamp register (96)", and then stores this difference as an offset value in the offset register (123);

an adder (125) that adds the offset value stored in the offset register (123) to a current value of the timestamp counter (94") and then re-loads the sum into the timestamp counter; and,

a second comparator that compares the current timestamp value stored by the timestamp counter (94") with a prescribed reset value and, upon detecting a coincidence, sets the sync flag latch (102") to indicate that a next timestamp value correction is to be made, in response to a next first command byte.

15. The network as set forth in Claim 14, wherein the last timestamp value transmitted by the control node corresponds to the timestamp value stored in the timestamp register (96') of the control node at the time that the previous command byte was transmitted.

16. The network as set forth in Claim 14, wherein the medium-access control subsystem (51) uses a reservation-based TDMA protcol.

17. The network as set forth in Claim 14, wherein any selected one of the nodes in the network can be assigned to serve as the control node at different times.

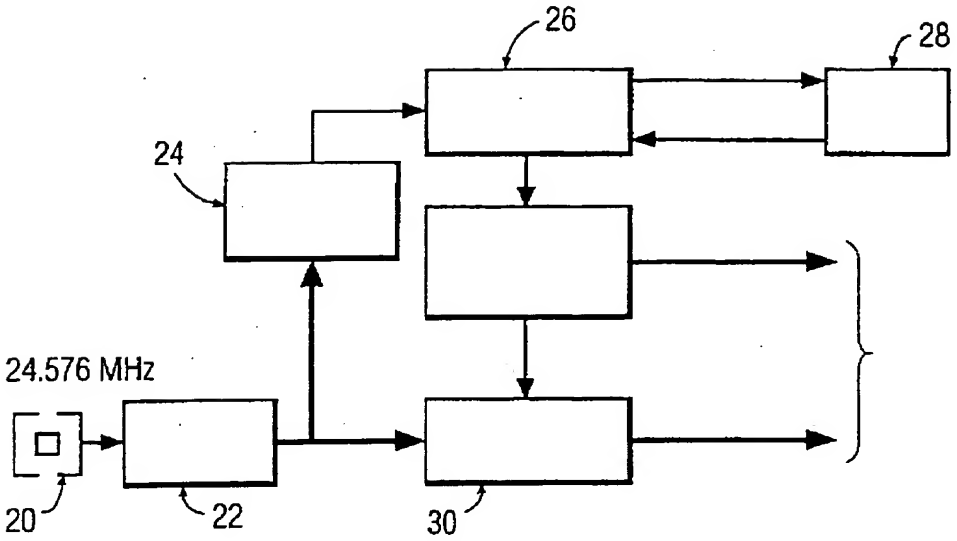


FIG. 1

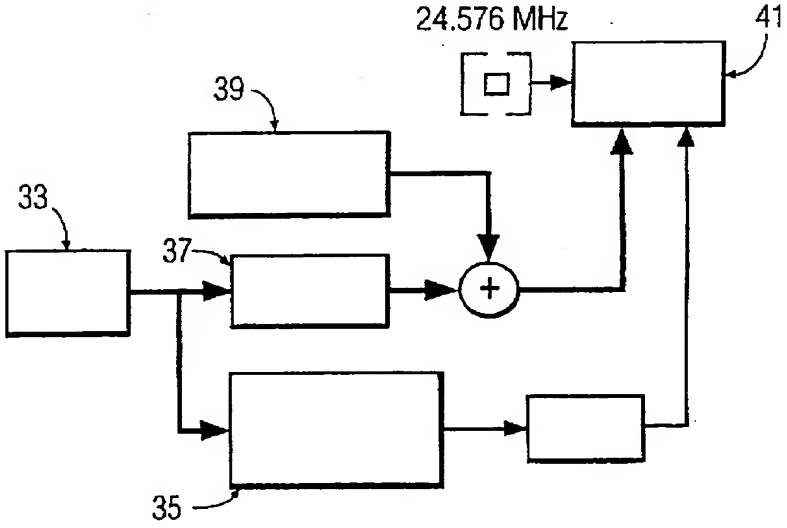
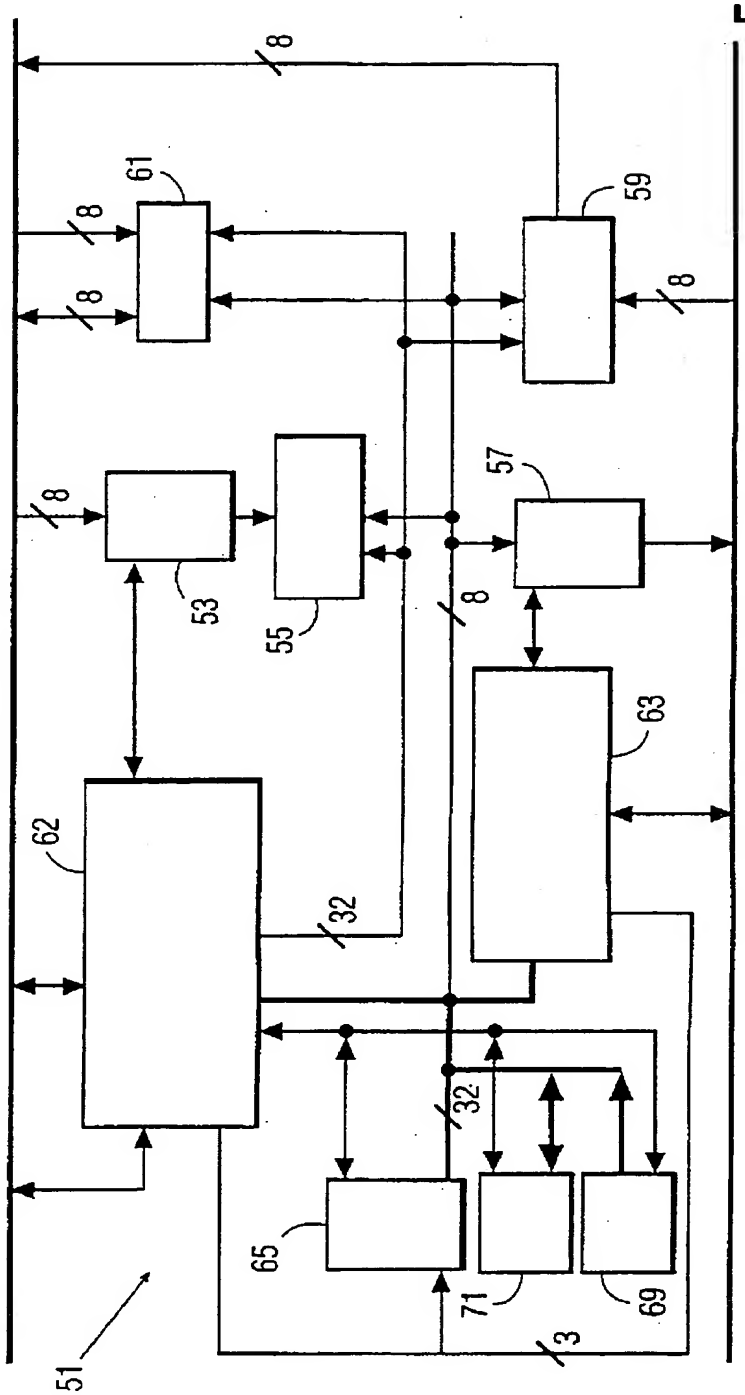


FIG. 2

FIG. 3

FIG. 4



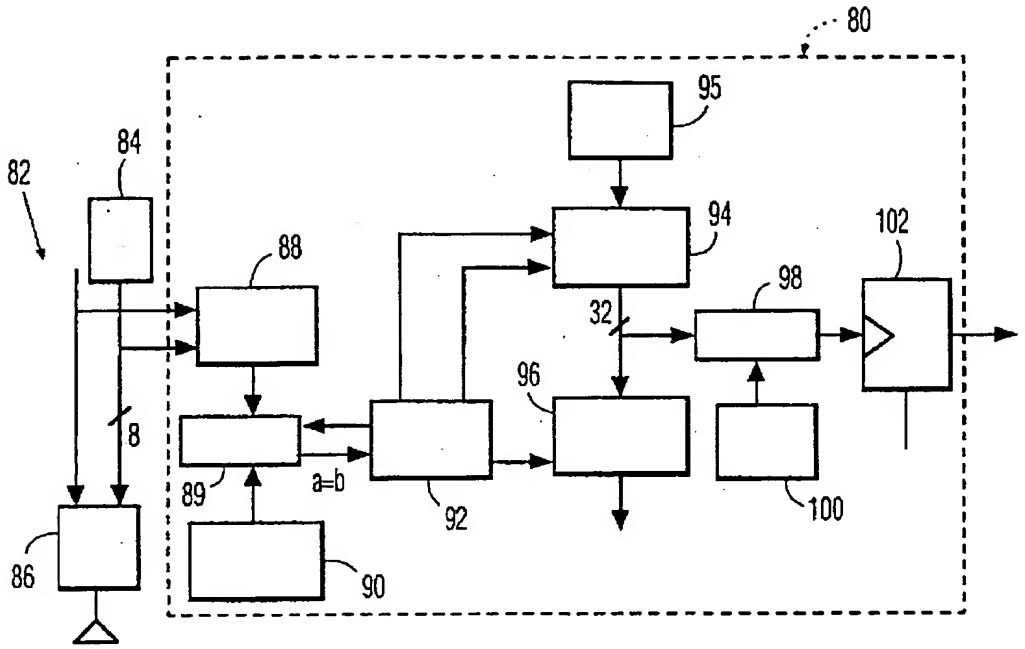

80
95
84
82
94
102
88
98
32
96
8
89
a=b
92
100
86
90

FIG. 5

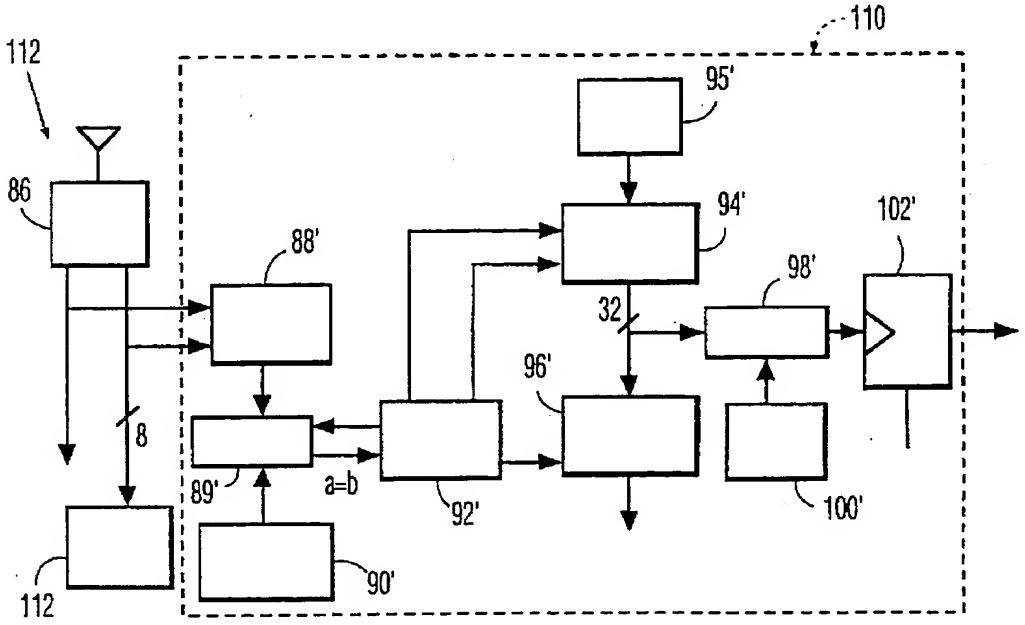

110
112
95'
86
94'
102'
88'
98'
32
96'
8
89'
a=b
92'
100'
112
90'

FIG. 6

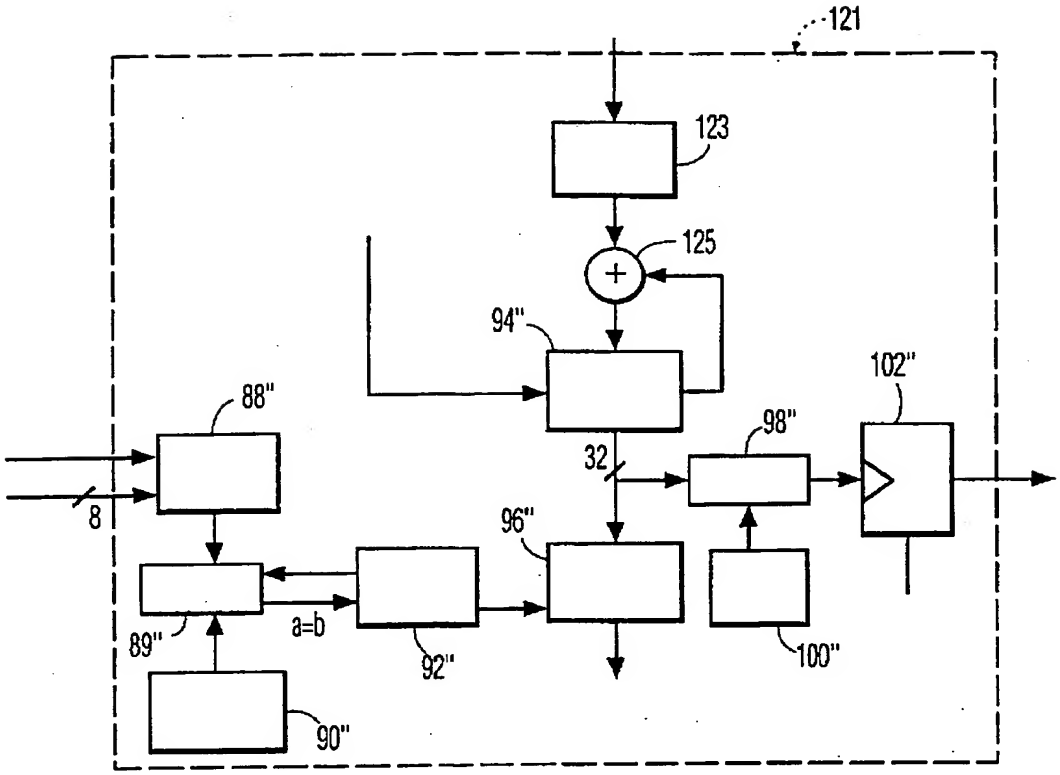


FIG. 7